ARTICLE IN PRESS

Future Generation Computer Systems ▮ (▮▮▮▮) ▮▮▮–▮▮▮

Contents lists available at ScienceDirect

Future Generation Computer Systems

journal homepage: www.elsevier.com/locate/fgcs

Applications of provenance in performance prediction and data storage optimisation

Simon Woodman *, Hugo Hiden, Paul Watson

School of Computing Science, Newcastle University, Newcastle upon Tyne, UK

HIGHLIGHTS

- Comprehensive provenance information can be captured from cloud based services.
- Provenance can be represented efficiently in a graph database.
- Intermediate data can be automatically re-generated from the provenance graph.
- Provenance graphs fused with performance data can predict future execution times.
- Data storage can be optimised based on the real world cost of data reproduction.

ARTICLE INFO

Article history:
Received 13 May 2016
Received in revised form
29 September 2016
Accepted 7 January 2017
Available online xxxx

Keywords:
E-science
Workflow
Provenance
Machine learning

ABSTRACT

Accurate and comprehensive storage of provenance information is a basic requirement for modern scientific computing. A significant effort in recent years has developed robust theories and standards for the representation of these traces across a variety of execution platforms. Whilst these are necessary to enable repeatability they do not exploit the captured information to its full potential.

This data is increasingly being captured from applications hosted on Cloud Computing platforms, which offer large scale computing resources without significant up front costs. Medical applications, which generate large datasets are also suited to cloud computing as the practicalities of storing and processing such data locally are becoming increasingly challenging.

This paper shows how provenance can be captured from medical applications, stored using a graph database and then used to answer audit questions and enable repeatability. This static provenance will then be combined with performance data to predict future workloads, inform decision makers and reduce latency. Finally, cost models which are based on real world cloud computing costs will be used to determine optimum strategies for data retention over potentially extended periods of time.

1. Introduction

The capture, storage and analysis of provenance traces are traditionally exploited to give users the answer to questions such as 'What algorithm was used to generate a particular result?'. The result can be visualised in the form of a directed graph that shows how data, workflows and services combined to create the data that is of interest. In scientific and medical research, this information is important as it explains the process used to generate the result (the data) and so allows others to judge its value and whether or not best practices have been followed.

A typical provenance trace can be represented as a graph structure tracing the flow of information from its source, through any processing and analysis steps and onto its final form. In addition to capturing this basic task dependency, the directed graph structure can be augmented with additional properties that represent the configuration of algorithms, data identifiers, code and library versions etc.

Being able to answer this class of question is important when processing sensitive information such medical data as it enables a level of operational auditing which can ensure that all data has been processed consistently.

The inverse can also be addressed: 'What results have been produced using a certain algorithm'. The answer to this question is also particularly relevant in medical applications as it allows results to be regenerated if enhanced versions of algorithms become available or if code is subsequently found to contain errors (an application of such a situation is presented in Section 3.3, where updated versions of predictive models were developed several times during the course of the study).

* Corresponding author.
E-mail address: simon.woodman@ncl.ac.uk (S. Woodman).

http://dx.doi.org/10.1016/j.future.2017.01.003

Please cite this article in press as: S. Woodman, et al., Applications of provenance in performance prediction and data storage optimisation, Future Generation Computer Systems (2017), http://dx.doi.org/10.1016/j.future.2017.01.003

In addition to allowing basic questions regarding both the generation of data and its subsequent use to be answered, there is also the potential develop data processing environments that make use of provenance traces to enable higher level operations to be automated. For example, a workflow platform may make use of provenance data to reconstruct "virtual workflows" [1,2] allowing the re-generation of results or intermediate data even if the original workflows that produce them have since been deleted. These virtual workflows can also be used to re-generate results if some of the intermediate processing steps need to be altered (for example to update analysis code or to correct erroneous algorithms [3]).

The use of data analytics and associated cloud resources has the potential to revolutionise the treatment of a number of chronic conditions. However, the volumes of data collected are frequently extremely large and analysis can require significant computational resources. Cloud computing offers a convenient solution as it can be paid for as needed and is capable of scaling to store and process large numbers of datasets simultaneously.

However, because the charging model for the cloud represents, to some extent, an unknown cost and therefore risk to project managers, it is important to have an estimate of the likely data processing and storage costs that will be required to perform a given set of experiments/study. If, in addition to basic provenance data, information regarding execution time, computation resources required and intermediate data sizes is collected, there is the potential to develop predictive models of cloud processing costs to address this issue.

The cost of execution is not the only cost associated with data analysis—the original raw data and results must also be stored. For large cohort medical studies the cost of data storage will be significant especially given that funding bodies mandate that all data must be retained for a considerable period following the end of the project.[1] An issue, particularly for medical workloads is that this data may be very large, and in some cases, the intermediate data generated during the execution may be orders of magnitude larger still [4]. At one extreme, it is trivial to calculate the cost associated with retaining all data basic on cloud providers published pricing information. At the other extreme, it is also possible to regenerate the data (subject to certain conditions) by constructing and executing a 'virtual workflow' to regenerate the data, thereby retaining none of the intermediate data. Clearly these offer two opposing options for being able to obtain the data at a later point in time. Further, it is possible to optimise the storage strategy by combining these two approaches—storing some datasets and regenerating others. These strategies can be compared by applying cost models based on current cloud providers storage and compute offerings.

The questions outlined above and platform requirements necessary to address them are summarised in Table 1. Almost all of the questions rely on storing the structure of the provenance graph and analysing it to produce a set of relevant traces or datasets. Some of the questions, those involving predicting future execution time also rely on having sufficient historical provenance and performance data to be able to construct a reliable model. Finally, to answer some of the questions it is also necessary to be able to predict future storage costs.

This paper will use a real world case study from a collaborative project with colleagues who are interested in the links between levels of physical activity and diseases such as Type II diabetes and cardiovascular disease. This case study, and others we have experienced in the medical sciences, are interesting as they exhibit interesting characteristics such as large volumes of data, algorithms which are frequently updated and long running studies. Other domains exist which exhibit some or all of these characteristics and would be good candidates for applying the techniques presented here.

Section 2 of this paper presents an introduction to physical activity analysis—one of the medical case studies that is used through the remainder of the paper. Section 3 shows how provenance data can be captured and represented in a data model tailored for this domain. The provenance information is stored in a graph database and queried to allow a set of representative questions to be answered. Section 4 introduces the ideas of capturing performance data and combining this with the provenance store in order to predict future execution times of workflows. In Section 5 candidate data retention policies are introduced to determine whether or not it is better to store data or regenerate it on demand. These are combined with cost models from commercial cloud providers to determine the optimum retention policy. Related work is discussed in Section 6 and finally, conclusions are drawn in Section 7.

2. Physical activity analysis

Historical and ongoing research projects are investigating the links between levels of physical activity and chronic conditions such as Type II Diabetes and cardiovascular disease.[2,3] The most common method for monitoring the physical activity of a subject is via the use of wearable devices such as accelerometers. Typically these take the form of a wristwatch that captures data at 100 Hz over a period of several weeks. Devices such as the GENEActiv and Actiwatch are commonly deployed and a wide body of research has been published using data captured in this way [5].

Movement data takes the basic form of a long timeseries containing three distinct channels of data captured from three perpendicular axes (X, Y & Z). An analysis of the patterns within this data can give an insight into a number of underlying conditions for example Activity level [6], which can be used to indicate quality of life and aid in the assessment and measure rehabilitation of some medical conditions.

The issue, however, is that as the wrist worn accelerometers measure movement data over three axes at approximately 100 Hz for many days, the quantity of data to be analysed is large—a typical data file is approximately 800 MB in size and comprises some 100 million rows of data. Once collected, the analysis procedure [7] for this data involves categorising the acceleration signals into one of several categories for instance: Sedentary, Light Activity, Walking and Running. This is then used to make recommendations as to suitable exercise plans for that specific patient. Further analysis may also be conducted [8] which can answer different research questions such as whether the patients sleep is affected by a change in medication or investigate exercise patterns across a large population.

Although the task of processing data for a single patient is tractable, clinical studies have collected movement data from large numbers of participants and analysing this data requires larger scale facilities. For example one of the studies processed using our infrastructure contain 300 GB of data collected from 1000 participants. Even this volume of data is modest compared with the UK Biobank, which has collected 24 TB of movement data from 100,000 participants.

Clearly, given an hour of typical processing time for a simple analysis, the task of processing data from a complete study is

[1] MRC mandates that all research data must be retained for 10 years—if the data relates to clinical trials it must be retained even longer, effectively indefinitely.

[2] http://www.directclinicaltrial.org.uk/.

[3] http://optimistic-dm.eu/.

Please cite this article in press as: S. Woodman, et al., Applications of provenance in performance prediction and data storage optimisation, Future Generation Computer Systems (2017), http://dx.doi.org/10.1016/j.future.2017.01.003

Table 1
Requirements for provenance analysis and cost prediction.

	Graph structure analysis	Execution time prediction	Cost prediction
What algorithm was used to generate a result?	×		
What results have been generated by an algorithm?	×		
How can we apply an updated algorithm?	×		
How long will it take to analyse a study?	×	×	
How much work is in the system?	×	×	
How much will it cost to store the data for N years?			×
How much will it cost to regenerate the data?	×	×	×
What is the most efficient way to keep the data?	×	×	×

not a trivial one and raises issues regarding both the provision of appropriate resources to complete an analysis in an acceptable time frame and also keeping track of the analysis processes and software versions used. This latter requirement is particularly important as movement analysis algorithms are the subject of active research with improvements and fixes published frequently. For example, one popular algorithm [8] has seen thirteen releases in a two year period in order to fix bugs and increase functionality. The use of workflow engines to coordinate the analysis of data generated in these projects is increasingly common as they can mitigate some of these issues by providing suites of services used in traditional Extract, Translate, Load pipelines [9].

3. Provenance capture and storage

Typically, a provenance trace is represented as a directed graph whose nodes are either *data items* or *processes* that take in data items as input, perform a computation and produce data items as output. The graph's arcs link data to processes, indicating that an item of data was either produced by or consumed by a process. The graph can contain the sub-graphs of independent workflow enactments, linked together by data items that are common to more than one enactment—for example a data item that is generated by one workflow and consumed by others.

The analysis of such traces can be used to answer questions such as:

- Q1: Find all direct and indirect dependencies of data item D_i.
- Q2: Find all computations and data items used to generate D_i (ancestor query).
- Q3: Find all data items that are derived from data D_i, or from service S_j or workflow W_k (descendant query).
- Q4: If the computation that generated trace T_l is re-run, are the results the same?

Q1 is generally considered the baseline query that all provenance management systems should support, and some effort has been invested in ensuring its efficient processing [10,11]. This high level query is specialised into more meaningful queries by the Q2 and Q3 queries.

Queries of type 2 are typically used to explain the presence of a certain data item in the output, as they return the fragment of the input dataset that has contributed, directly or indirectly, to that output. This often has an intuitive interpretation in settings where the workflow maps data from one domain onto another.

Type Q3 queries are used to perform impact analysis, as they return all and only the data items whose value is influenced, directly or indirectly, by a certain input (or intermediate data located upstream in the provenance graph).

There are however restrictions on the questions that can be answered based on this trace information, and on how that information can be usefully exploited. Major issues surround the important concept of reproducibility.

Whilst capturing a provenance trace is necessary, it is not sufficient to allow a computation to be repeated as a situation known as *workflow decay* [12] can occur. The problem is that while provenance systems can store information on how the data was generated, they do not store copies of the key actors in the computation: the workflows, services and data. Even if a diligent researcher retains all their data and their calculation workflows, there remains the possibility that any services used may disappear or be modified rendering workflows that depend on them inoperable. To address this issue, systems such as the e-Science Central platform [13] retain all versions of data workflows and services used in any computations and make can make use of a provenance traces reconstruct any analysis workflows that may have been deleted.

By using these capabilities, we are able to answer Q4 and other queries relating to versions of services used within the workflow—if the versions of services change, what effect does this have on the results. As will be shown in Section 3.3 this can have effects on the analysis of medical data in two ways. Firstly, it allows an investigator to ascertain that the same version of the service was used throughout a long running study. Secondly, it allows the developers of analytics methods to compare newer versions of algorithms against older ones [14].

3.1. Capturing provenance

A number of standards exist for representing provenance traces such as OPM and PROV [15]. The model presented here and shown in Fig. 1, is based on the Open Provenance Model (OPM) Version 1.1, and can be used to produce a directed acyclic graph of the history of an object. Objects in OPM are categorised as either artefacts, processes or actors which correspond to nodes within the graph. Vertices in the graph represent relationships between two objects and are of types such as wasGeneratedBy, used, wasControlledBy and wasDerivedFrom. The OPM Core model has been extended with subclasses to identify the different types of processes and artefacts we are concerned with. For example, execution of a workflow and a service within a workflow have been differentiated. The relationship between workflow execution and service execution is of type *contained* is not strictly required by our model but its inclusion makes it easier to generate the different views of the process. The artefacts described in the model are also subclassed in order to differentiate between versions of data, services, libraries and workflows. The fact that the *User* controls all of the processes has been omitted from the diagram to aid clarity [3].

Our model deals with two different types of data artefact: *Data Version* and *Transient Data*. This is due to the semantics of workflows in e-Science Central: data generated by a workflow must be explicitly saved using a service which 'exports' the data back into the e-Science Central repository. Any data which is not explicitly exported will be discarded when the workflow completes.

In addition to storing the structure of the provenance graph in terms of processes, artefacts and the relationship between them, it is necessary to store the properties of the processes and characteristics of the artefact. For example, any parameters used

Please cite this article in press as: S. Woodman, et al., Applications of provenance in performance prediction and data storage optimisation, Future Generation Computer Systems (2017), http://dx.doi.org/10.1016/j.future.2017.01.003

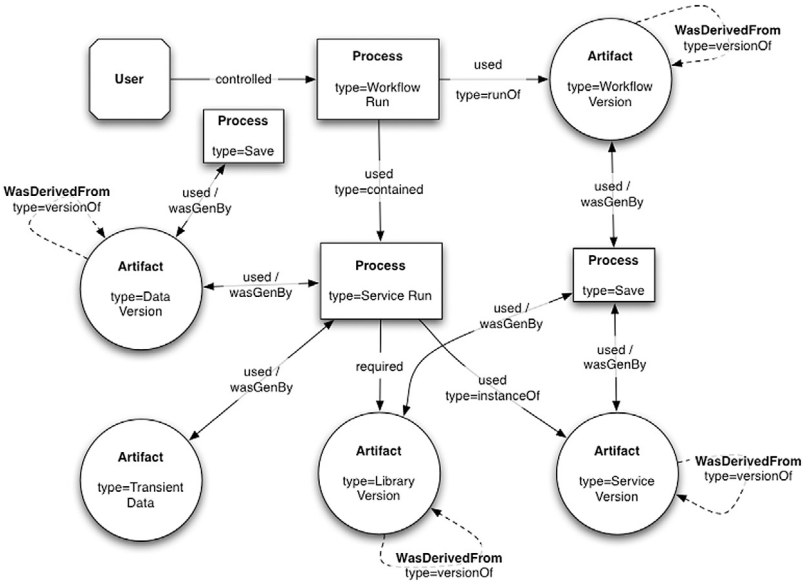

Fig. 1. e-Science Central provenance model.

in the process must be stored, and for the purposes of predicting future execution times, performance characteristics and execution time are also logged.

It is pertinent to point out that not all services are repeatable or will give consistent results. Services may be non-idempotent (for example a database INSERT operation) or they may include non-deterministic algorithms. e-Science Central attempts to account for this by calculating a hash of any intermediate data. This allows the detection of situations where re-running code produces different results. In these cases, if the intermediate data is critical, it can be persisted indefinitely within e-Science Central.

3.2. PAC1 activity analysis

The workflow used within the Newcastle 85+ project to analyse the data and determine the levels of physical activity is shown in Fig. 2. This example will be used throughout this paper (referred to as the PAC1 workflow) to illustrate the contributions being made. The workflow operates on binary data from a GENEA 1.0 activity monitor and the purpose of the workflow is to extract the data into the format required by the PAC1 algorithm—CSV formatted timestamp, x, y, z and then run the algorithm on that CSV data [7]. Other blocks in the workflow are concerned with managing the study—renaming files for consistency and attaching metadata from the header of the binary file. The output is a report which is exported from the workflow engine to the e-Science Central data store.

The ancestral provenance trace of the PAC1 report is shown in Fig. 3. The workflow contained 9 services and was controlled by the user with Id `Simon`. In a largely linear fashion services `used` transient data that `was generated by` the previous service in the chain. The deviation from this linear pipeline is that the rename files service used not only the report generated by the previous service but also the original binary file (in order to get the name to rename the report to). The properties of the graph have not been shown because of clarity and space limitations but data relating to identifiers, versions, timings and input parameters are all captured and stored along with the graph structure.

3.3. Using a graph database to store and interrogate provenance

Given that the provenance structure being stored is a directed acyclic graph, we chose to store it in the non-relational graph database, Neo4j[4] and various components of the e-Science Central platform can log provenance data in this format. Neo4j differs from traditional relational databases as its structure does not consist of tables, rows and columns, but instead of nodes, relationships and properties. This provides a much more natural fit to our model, and allows us to store the provenance graph directly instead of encoding it in a relational model. Neo4j has also been shown to scale well, and most importantly, to perform well in terms of queries even when storing a very large graph structure [16].

Instead of SQL queries, Neo4j supports a language graph query language, Cypher[5] and an operation known as a *traversal*. The latter is an imperative way of defining how to traverse the graph from a particular starting node—what types of nodes and edges to traverse, in what direction, and when to stop. Cypher gives a declarative way of matching patterns in the graph. Using either of these methods we are able to answer the high level questions identified earlier.

A project which made extensive use of Q3 type of queries was Limbs Alive [17] which tried to determine whether the rehabilitation following Stroke could be improved using computer games. During this project, one partner was developing a model which predicted a CAHAI score (a measure of upper limb movement frequently used in Stroke rehabilitation). As more data was collected from the patients a better model was able to be built. A traverser was used to construct a "virtual workflow" which was used to regenerate the results using the updated model and automatically compare them to the previous ones [17].

This section has shown how it is possible to model, capture and store the provenance of data generated by e-Science Central workflows. Real world examples from medical studies have been used to illustrate some of the concepts and the applications of provenance traces in generating virtual workflows. The following sections will show how this data can be used for other purposes including predicting compute requirements and optimising storage provision.

[4] https://neo4j.com/.
[5] http://neo4j.com/developer/cypher/.

Please cite this article in press as: S. Woodman, et al., Applications of provenance in performance prediction and data storage optimisation, Future Generation Computer Systems (2017), http://dx.doi.org/10.1016/j.future.2017.01.003

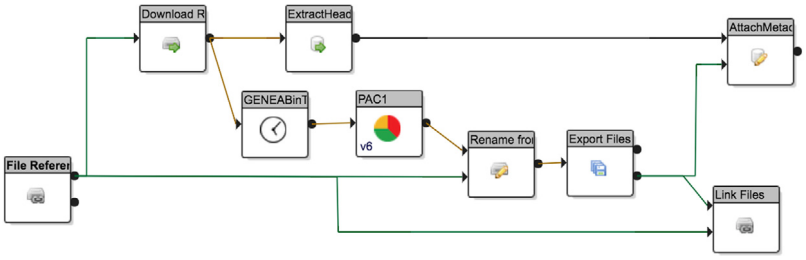

Fig. 2. PAC1 workflow for Newcastle 85+.

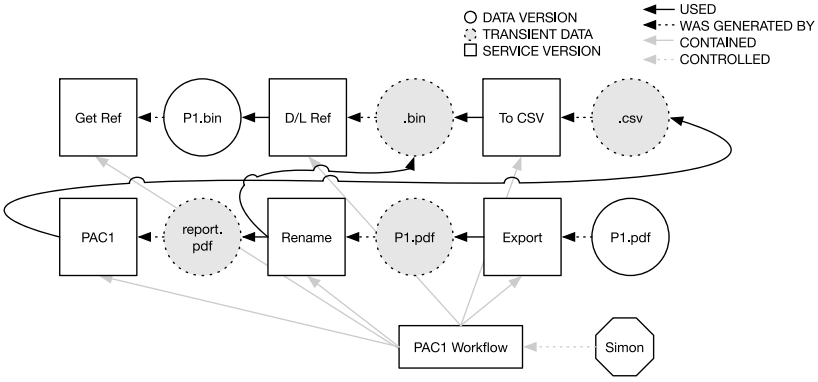

Fig. 3. Provenance trace for report generated by PAC1 workflow.

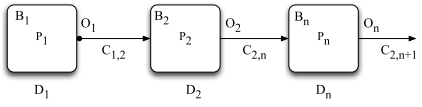

Fig. 4. e-Science Central workflow structure.

4. Predicting future execution times

One of the practical uses of provenance when married with performance data is to attempt to predict future execution times from historical performance of a given component. There are three key drivers for medical studies: returning an analysis in a predictable amount of time when there is a patient physically waiting for a diagnosis/consultation; estimating the compute resource required to support the analysis of an entire study within a given timeframe and estimating the resources required to re-process a set of data given an updated/corrected algorithm.

4.1. Modelling workflow execution times and data sizes

e-Science Central, workflows can be considered as a linked set of individual software components (blocks) which act sequentially upon items of data.

The basic structure of an e-Science Central workflow is illustrated in Fig. 4 which shows a number of connected blocks ($B_1 \ldots B_n$). Each of these blocks can contain a property set that defines its behaviour ($P_1 \ldots P_n$). The analysis pass of the workflow execution process will identify B_1 as the single data source block. The execution thread will first execute B_1 using the property set, P_1. This will take a period of time, D_1, and produce a piece of output data, O_1. This data item will be propagated to the second block in the workflow, D_2, along the connection, $C_{1,2}$. The second block, B_2, will then be executed using its property set, P_2, and input dataset, O_1. This process will take D_2 seconds.

From this it can be seen that the total actual execution duration for the workflow, D_{wf} can be expressed as:

$$D_{wf} = \sum_{i=1}^{n} D_i.$$

To generate an estimated execution duration for the workflow, $\hat{D_{wf}}$, a summation of duration estimates for the individual workflow blocks is therefore required:

$$\hat{D_{wf}} = \sum_{i=1}^{n} \hat{D_i}.$$

This approach is applicable to the e-Science Central workflow engine because it does not attempt to execute any blocks in parallel, so the total execution time can easily be calculated. For cases where workflow paths can be operated in parallel, the longest duration for each parallel path must be summed in order to predict the total execution time. In order to generate a prediction of the execution duration for a particular block, $\hat{D_i}$, a relationship needs to be defined that relates execution duration to the various attributes of the block that can influence performance. In general the execution duration for a block will be a function of the input data size to the block, O_{i-1}, the actual code within the block and the block parameter settings, P_i. The estimated duration of any block within the workflow can therefore be calculated using:

$$\hat{D_i} = f_{Di}(P_i, O_{i-1})$$

where f_{Di} represents the predictive duration model for the ith workflow block. From this, it follows that the total workflow duration can be predicted by:

$$\hat{D_{wf}} = \sum_{i=1}^{n} (f_{Di}(P_i, O_{i-1})).$$

Because the duration estimate for each block within the workflow is dependent upon the size of the data flowing into it, the process of estimating the duration of a multi-block workflow is complicated by the fact that, for non data source blocks (i.e. most blocks within the workflow) a value for the input data size must also be estimated. If the output data size for a block is assumed, like the duration estimate, to be a function of the input size (O_{i-1}) and the block settings (P_i), the output data size for a given block within a workflow can be modelled using:

$$\hat{O_i} = f_{Oi}(P_i, \hat{O_{i-1}})$$

Please cite this article in press as: S. Woodman, et al., Applications of provenance in performance prediction and data storage optimisation, Future Generation Computer Systems (2017), http://dx.doi.org/10.1016/j.future.2017.01.003

where f_{Oi} represents the predictive output size model for the ith workflow block. During the process of producing a duration estimate for an entire workflow, this size estimate is propagated throughout the workflow in place of the actual data sizes. It follows, therefore that as the size of the workflow increases, the model prediction will be degraded by both the errors in predicting the duration of each block and also the errors accumulated by propagating size estimates to each duration prediction. The availability of accurate models which can predict the quantity of data produced by executing individual blocks is therefore central to accurately estimating total workflow execution time.

In order to capture performance data, the e-Science Central workflow engines send data regarding the total execution duration, the volume of data processed and configuration properties for all blocks in a workflow to the provenance server asynchronously via a message queue, a pattern adopted to reduce the impact of data logging on the primary e-Science Central database.

4.2. Model types

The relationship between block duration and observed execution data for blocks within an e-Science Central workflow can fall into one of three broad categories:

1. The block duration can be estimated using a linear combination of the execution data contained within the performance database.
2. The block duration follows a non-linear relationship between execution time and the captured performance data.
3. The block duration exhibits no correlation to any of the observed execution data.

The performance modelling system can maintain models for each version of each block observed during workflow executions and generate duration predictions using the most appropriate model on demand. This requires models to be managed (Section 4.3) and also the facility to generate some sort of prediction even in situations where the quantity of observed data is insufficient to create one of the models described above (Section 4.3).

4.3. Model management

One of the key requirements for the performance modelling application is to provide a robust prediction of performance properties that are refined as more data becomes available. Therefore, a number of fallback predictions are provided to cater for situations where models are unavailable for a block:

1. If there is no model available for a specific version of a block a version agnostic model is used.
2. If there are no models of any sort for a block, but there is at least one observation for a block, average values for execution duration and output size will be used.
3. If there is no data of any sort for a block, the average duration for all blocks will be used and the average output data size will be used for predicting output sizes.

The reasoning behind the above logic is to return a prediction wherever possible and to always return the best prediction that the system can provide at a given point in time. In addition, because the nature of a block cannot generally be determined *a priori*, the performance modelling system must be able to determine automatically whether the execution duration of a block is linear, non-linear or uncorrelated with respect to the observed data. In order to achieve this, the system builds every type of model contained within its library for each block. This pattern has been adopted for some earlier chemical modelling work [18] and, once built, the model demonstrating the best performance on a set of test data is used to generate duration predictions until the next model update step.

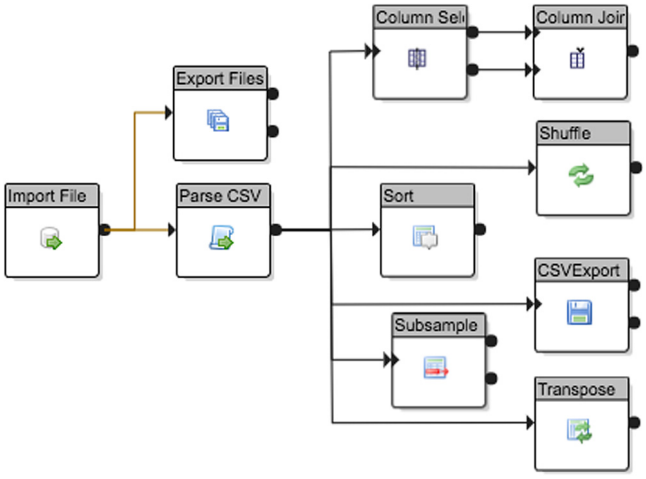

Fig. 5. Simple data generation workflow.

4.4. Modelling simple workflows

In order to demonstrate the suggested approach to modelling workflow performance, simple workflows containing a set of trivial blocks under ideal conditions were modelled to investigate whether it was indeed possible to generate reliable performance models. These experiments were performed in the Amazon AWS cloud in order to best represent the conditions under which the medical data processing workflows are executed in the other examples presented in this paper.

The workflow studied in this initial experiment (Fig. 5) contained ten simple Java blocks that are provided with every installation of e-Science Central. These blocks perform basic data manipulation tasks and are therefore more IO than CPU intensive. As such, the execution of these blocks is likely to be very highly correlated to the data volumes being passed through them.

All models built during these experiments were compared using the Root Mean Squared Error measurement (RMSE), and the correlation (r^2) between the predicted and observed execution durations. The workflow shown in Fig. 5 was executed 250 times with a set of randomly generated input files ranging in size from 7 kB to 14 MB. These files were pre-generated and one was selected at random for each of the 250 executions. The results of this experiment demonstrated that for the majority of blocks, it was possible to generate an accurate prediction of execution time based upon the size of the input data and the captured block configuration parameters. For example, Fig. 6 shows the duration prediction model for the Shuffle block (RMSE $=$ 1.329, $r^2 =$ 0.999).

In order to assess the performance of these models when applied to different workflows (containing a subset of the blocks shown in Fig. 5), a different workflow was constructed which again processed a set of randomly generated data. This new workflow was executed multiple times and for each execution, the actual duration was recorded along with a duration estimate generated by the performance monitoring system. The results are displayed in Fig. 7 and indicate RMSE $=$ 4.077 and $r^2 =$ 0.997 [19].

4.5. Modelling the PAC1 workflow

Following the same approach, the PAC1 workflow (Fig. 2) was modelled using the data capture architecture described above. Whilst data regarding tens of thousands of PAC1 workflow executions have been captured from the NE85+ and other studies, the vast majority of executions were for datasets of exactly the same size, which do not produce data relevant to the generation

Please cite this article in press as: S. Woodman, et al., Applications of provenance in performance prediction and data storage optimisation, Future Generation Computer Systems (2017), http://dx.doi.org/10.1016/j.future.2017.01.003

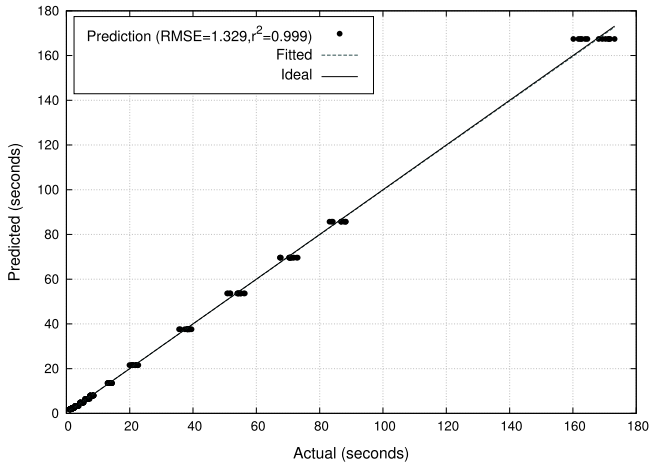

Fig. 6. Prediction of execution duration of the Shuffle block on Amazon EC2.

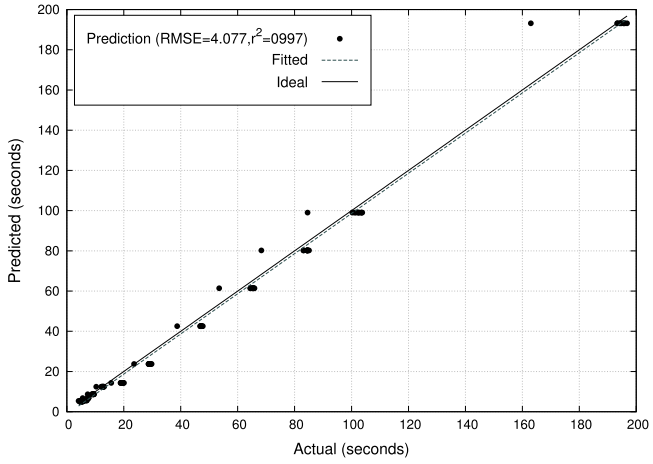

Fig. 7. Execution duration prediction for new workflow.

of regression models. However, within this data set there are some differences in input file sizes. This is due to truncated datasets, episodes of non-wear and other factors which resulted in devices being worn for less than the full two week period. Using this data, we were able to model execution duration using a variety of input data sizes ranging from 66 to 800 MB. The resultant model, shown in Fig. 8 demonstrated an r^2 of 0.99 and RMS error of 158.08. Given these results, we can be relatively confident that it is possible to predict the behaviour of the PAC1 workflow under a range of different input conditions. Additionally, taking these results into account, we can estimate that the average time taken for PAC1 to process a standard dataset is approximately 45 min.

This exercise, therefore, has demonstrated the feasibility of using provenance and performance data to estimate the resources required (both in terms of CPU time and also data volumes) to execute a real world workflow.

5. Optimising data storage

The studies described in this paper can generate not only significant amounts of primary, source, data but also lots of intermediate and output data from the processing and analysis. Most of the physical activity monitors used store data in a compressed, proprietary, binary format. This contrasts to the developers of the analysis toolboxes who typically want to remain device agnostic and so operate on data formats that are open and human readable (in the case of CSV) but less efficient in representing the data. A key component of any data analysis

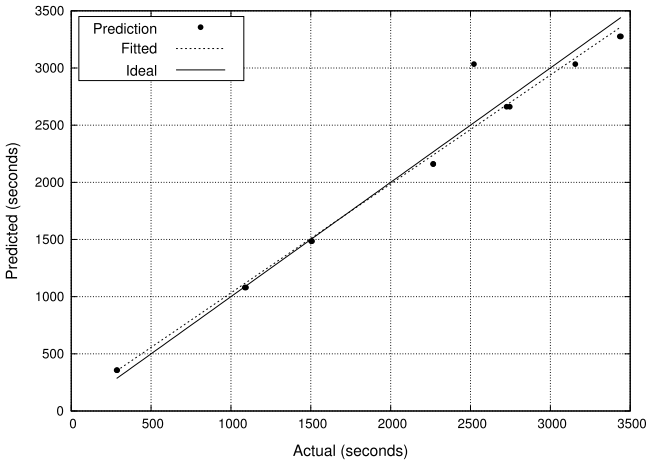

Fig. 8. Execution duration prediction for PAC1 calculation block.

workflow, therefore, will be to transform the data from the format it was collected in to the format required by the algorithm implementation.

In order to follow best practices for reproducible science, and increasingly commonly for funding or regulatory reasons, it is necessary to retain all project data for a prolonged period of time. One obvious solution is to retain all intermediate data in a repository that expands each time any new work is performed—an approach which has been demonstrated in the Taverna workflow system [20]. This approach is guaranteed to work as one has confidence that any intermediate result will always be immediately available. The evident drawback, however, is that such a provenance store has an unbounded size that continuously grows regardless of whether the cost of retaining a piece of intermediate data exceeds the cost of re-generating it. If, however, a complete provenance trace is stored, in many cases it will be possible to re-generate any piece of data using exactly the same process that was deployed in the original processing work. The drawback of this approach is that sometimes the computational cost required to regenerate results may exceed the cost of merely storing these results.

In order to effectively provide a comprehensive provenance storage system which can strike a balance between retaining and recalculating intermediate results, a cost model which can incorporate knowledge of storage-vs-computation costs is required. The decisions as to what to retain and what to re-generate are therefore a trade-off in terms of the long term storage cost and the cost (in terms of computation time) for re-generation for each given piece of data [21].

Let us formalise the provenance model described in Section 3 to define that an Activity 0, A_0 operates on Entity 1, E_1, to generate Entity 2 E_2 such that $E_2 = A_0(E_1)$. Given a provenance aware data store that was required at some point in the future to be able to produce any entity, a simple strategy would be to simply store these entities in perpetuity. However, given the system described in Section 3 that captures the provenance data it is possible to use this information to regenerate an exact copy of any piece of data on demand. In generating any piece of intermediate data, it is also necessary to account for the cost/time for the regeneration of any upstream data which may have been deleted.

When considering which pieces of data to retain in the provenance store and which to discard in favour of regeneration, it is possible to generate a set of candidate options which include every variation of keeping (K) or regenerating (R) an entity by re-executing the upstream activities. In general, the number of storage options is 2^n, where n is the number of blocks within the workflow. In reality, however, some of these candidate options are not feasible. There are a number of reasons for this:

Please cite this article in press as: S. Woodman, et al., Applications of provenance in performance prediction and data storage optimisation, Future Generation Computer Systems (2017), http://dx.doi.org/10.1016/j.future.2017.01.003

Input data In some cases, it is not possible to regenerate data. This occurs for example when some data is created by a process that is not under the control of the workflow engine. Typically, this occurs when data is uploaded by a user. We consider data in the system that does not have any incoming provenance as such an edge case.

Non deterministic and non idempotent operations As described in Section 3 some processes either have side effects or elements of non-determinism in their computation. The output of such services must always be retained in the storage system.

The number of valid candidate policies is expressed in terms of the number of activities in the graph rather than the number of entities. This is because in order to regenerate an entity the activity must be re-executed. If an activity generates multiple entities they will all be regenerated but it is the individual entity that we are interested in. Candidate policies which do not keep all of the outputs of an activity are still valid—the entities may be of unequal size and so have different implications for storing them. In general, the number of valid candidate solutions to the problem of optimally storing provenance for a given workflow is:

$$N = 2^{n-(i+d+m)} \tag{1}$$

where: `n` is the total number of services, `i` is the number of input data services, `d` is the number of non-deterministic services and `m` is the number of non-idempotent services.

5.1. Storing and regenerating entities

Given a set of valid candidate retention policies for the data (entities) generated as part of the workflow execution we can compute storage and compute requirements associated with adopting each retention policy. This "logical" cost can then be mapped onto a monetary cost.

The following must be considered when calculating the logical cost of adopting each policy:

Storage costs The cost of storing an entity in Cloud storage such as Amazon S3 or Azure Blob Store for one month.

Regeneration costs the cost of the compute time to enact the activities necessary to regenerate the entity. This will be expressed in terms of CPU hours (CPUh).

Fixed costs The costs of maintaining any underpinning systems such as application servers, databases, workflow engines. We will ignore this cost as it is assumed to be fixed across all retention policies.

5.2. Regeneration process (entity regeneration cost)

In order to regenerate an entity, the system will need to execute the activities between the 'last' available entity and the desired entity. By 'last' available entity we mean the closest 'upstream' entity in the provenance graph. Without branches, this is the sum of the execution times between an entity, E_n and the last stored entity, E_j.

$$compute\_time_n = \sum_{i=j}^{n} execution\_time(A_i). \tag{2}$$

Given a more complex structure where branches exist, the equation shown above can be generalised into one which includes the cost of regenerating the entities for the activities along the different branches involved in the generating of entity E_n. We define `nda` as the number of distinct activities along the branches leading to E_n.

$$compute\_time_n = \sum_{i=0}^{nda} execution\_time(A_i). \tag{3}$$

As before, we must include the storage impact of retaining k entities which are being kept rather than regenerated.

$$storage\_volume = \sum_{i=0}^{k} storage\_cost(E_i). \tag{4}$$

The total resource requirement is the sum of the compute time to regenerate each of the entities which have not been kept and the storage volume of those which have, where there are p entities which need to be regenerated

$$policy\_cost = \left(\sum_{i=0}^{p} compute\_time_p\right) + storage\_volume. \tag{5}$$

The ERC policy gives accurate representation of the costs of regenerating individual items but it is also possible to consider regenerating all entities at the same time, the Global Regeneration Cost which will require different amounts of compute and storage. Other, more complicated policies can be applied which, for example, introduce a maximum latency that data must be regenerated within to ensure that data is available in a timely fashion.

The PAC1 workflow as shown in Fig. 2 contains 9 activities. Two of the salient points of this process are that the conversion to CSV produces a large file which is approximately 4.5× the size of the compressed input data, and the PAC1 algorithm is memory intensive, requiring approximately 10 GB of physical RAM (see Table 2).

In this process, most activities are repeatable but the blocks which deal with obtaining the input data, are not. Thus, we need to keep the output of Get and Download references. Additionally, the Link Files activity is non-idempotent.

Therefore, from the original 2^9 candidate retention policies for this provenance trace, given the three activities mentioned above are not repeatable, this set can be pruned to 64 valid candidates.

5.3. Modelling future storage costs

Although one could assume that the cost of cloud storage will remain a constant in the future, Kryder's Rate [22] demonstrates that storage, as with most other computer components, is becoming cheaper to produce and is able to store information at higher densities. When we are projecting costs 10 years into the future, the potential storage costs can have a significant impact on the overall price of each retention policy. Thus our use of a constant rate would appear a little naïve. In order to adjust for this we have looked at historical price data for the Amazon S3 service (Glacier has remained a constant $0.01/GB/m since introduction and S3 Reduced Redundancy historical pricing is not available). The cost of S3 is shown in Fig. 9 and shows the price dropping over time. Notably, the price as of Summer 2015 is 1/5 of the cost when it was first introduced.

Although price reductions have historically occurred in stepwise manner, it is impossible to predict these steps in the future and we therefore project future storage costs as, on average, dropping at 1.6% per month.

It would be possible to also model the falling price and increasing speed of compute as well as the falling storage cost. However, this was not included in the model for a number of reasons: firstly, the characteristics of each computation are

Please cite this article in press as: S. Woodman, et al., Applications of provenance in performance prediction and data storage optimisation, Future Generation Computer Systems (2017), http://dx.doi.org/10.1016/j.future.2017.01.003

Table 2
Performance stats for PAC1 process.

ID	Activities	Duration (min)	ID	Entities	File size (GB)
A_0	Get file reference	0.05	E_0	File reference	0.0001
A_1	Download ref	2	E_1	Bin file	0.735
A_2	Extract headers	1	E_2	Header info	0.0005
A_3	Convert to CSV	10	E_3	CSV file	3.2
A_4	PAC1	45	E_4	Report	0.001
A_5	Rename report	0.2	E_5	Report	0.001
A_6	Export files	0.2	E_6	Report ref	0.0001
A_7	Attach metadata	0.2			
A_8	Link files	0.2			

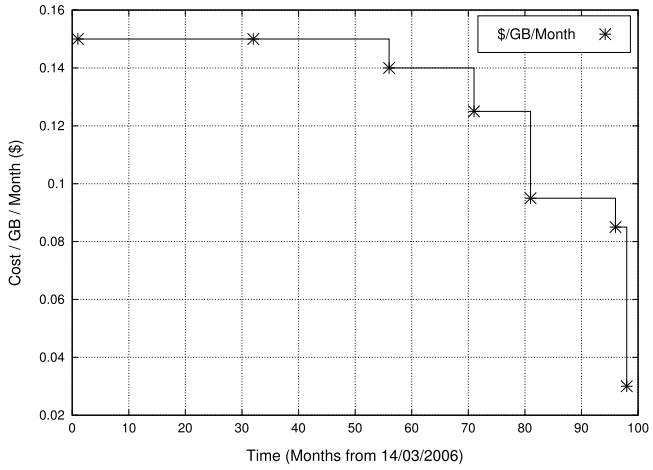

Fig. 9. Historical costs of Amazon S3.

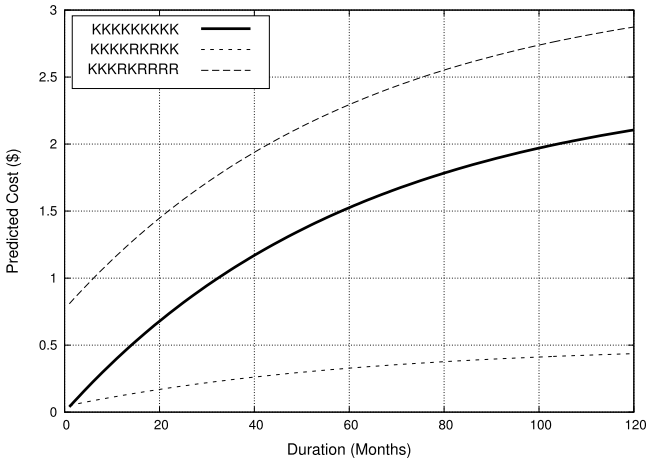

Fig. 10. PAC1 storage and regeneration costs.

different with some bound by different limits—CPU, memory, i/o etc. The increase in performance of EC2 machines would make it hard to generalise about the speedup in computation for different problems. Secondly, many of the computations we have experience of are inherently single threaded. Thus, the addition of more CPU cores will not affect the length of the computation (although it *could* reduce the cost if a lower powered machine were able to do the work in the future). The fact that storage is constant with the only variable of price means that it is safe to include in our model. It is likely that the ratio between increased performance/lower cost of compute and the storage cost per GB will make some candidate policies more or less attractive but it is impossible to predict in what way. The uncertainty over the compute costs for individual applications was the reason for its omission. When we apply the file size and duration data to the valid candidate policies we can see that keeping all of the data would cost $2 over a ten year period (Shown in Fig. 10). The most expensive option is KKKRKRRRR at $2.86 so keeping all of the data is not the worst case scenario. However, the best case scenario using the ERC policy is KKKKRKRKK which will cost just over $0.4 to store or regenerate over a ten year period. Looking at the *shape* of the workflow the numbers presented are not surprising. The policies which are more expensive favour regenerating the report and storing the large intermediate CSV file. The cheap policies (the distribution of policies is uneven with most costing around $0.4 or $2.8) favour the opposite, to keep the small report and regenerate the large CSV file if required, a more natural choice. Further, whilst most of our calculations have assumed regenerating the data a single time, this example shows us that we could regenerate everything four times and it would still be cheaper than storing the entire dataset.

Although the numbers in this example are small (even the most expensive policy will only cost $2.80 to store the data for a 10 year period), it should be noted that this is for a single execution of a single workflow. Given that the Whitehall study has 4.3k participants and each dataset collected has been processed using this workflow the cost savings are much more significant. Studies such as the UK Biobank with 100k participants will demonstrate an even larger saving.

6. Related work

Research on provenance capture, storage and visualisation, particularly in the area of e-Science has been an active area for many years [23]. Much work has been done on how provenance can be captured from heterogeneous systems such as workflow systems and databases and subsequently integrated and reasoned upon [24,25]. Recently the growth of non-relational, and particularly graph databases have been adopted as provenance stores given that a provenance trace is, inherently, a graph structure [3]. Such databases offer both a natural domain fit for the storage of provenance and powerful query interfaces to interrogate and interpret provenance traces.

The focus of this paper is on some of the more practical uses of provenance that is collected in scientific workflow management systems. One of the frequent uses that we have not considered is the use of provenance for debugging applications which are not performing in the expected way [26]. Being able to view the state and parameters of successful vs failures can be a very useful feature in both debugging and optimising applications. Other work shows that it is possible to verify that a process did adhere to the contract (the retrospective provenance matches the prospective provenance) which is critical in industries subject to strict regulation and audit [27].

Taverna [28] and Kepler [29] are extremely popular workflow engines used in the scientific domain and capable of capturing provenance data. In many cases the facilities they offer are far greater than e-Science Central in terms of querying and exporting the provenance graph [30] and identifying collections used in computations [31]. However, e-Science Central is capable of

Please cite this article in press as: S. Woodman, et al., Applications of provenance in performance prediction and data storage optimisation, Future Generation Computer Systems (2017), http://dx.doi.org/10.1016/j.future.2017.01.003

dealing with service versions in a more graceful manner and, as this paper has shown, creating executable workflows from provenance traces.

Neo4j has previously been used by Wendell to capture provenance data about the software development process [32] and by Tylissanakis to store the provenance of scientific workflows [33]. The latter is similar to our work although it focuses on coordinating Web Services and does not consider the issue of versioning. However, they do mention briefly that the are able "reproduce simulation results" but do not go into detail.

The work presented by Duan [34] is of particular interest as it closely resembles ours but focuses on Grid deployment scenarios. One of the key differences is our use of the 'Panel of Experts' pattern [18] to generate multiple predictive models of each service rather than their use of a Radial Basis Function neural network. We have found that it is imperative to include multiple modelling techniques as some components will model significantly better or worse depending on the technique used.

The work by Cushing [35] discusses how to scale Map-Reduce style problems based on the expected execution time. The aim here is to reduce the overall computation time by dedicating more resources to components which are expected to take a longer duration. In addition they aim to prevent starvation of future components due to a previous one having not completed execution. We do not restrict our execution pattern to Map-Reduce, although we are able to construct such a pattern using e-Science Central workflows.

Within the context of cloud computing, Roy [36] use autoregressive moving averages to predict the current workload of a system. However, they are concerned with scaling cloud architecture to minimise response time in a web application rather than scientific workflow applications which exhibit different characteristics.

The literature around the Prophesy system details some approaches to leveraging multiple predictive models to generate a prediction for a larger unit of work [37]. As their work is principally aimed at lower level functions with more complex inter-relationships they generate what can be considered to be a cross-product of model relationships between each 'kernel' of computation. Our approach differs in that we only consider the effects of the data transferred from one component to another and, given that we are dealing with higher level components without such inter-relationships, we do not need to compute the cross-product of all components. We also show that it is feasible to use the output of one predictive model as the input to another whereas other systems simply consider the summation of the predictions from each model [38].

Typically systems which capture provenance do not attempt to also capture the raw data itself. PBase [39], which is based on the ProvONE model for interoperability, offers a web UI for user query and interaction. Early versions were based on an RDF store implemented in the Apache Jena framework but more recent versions have used the Neo4j graph database. The optimisation techniques presented in this paper could be applied to query the PBase system in order to optimise a third party data store.

Provenance aware storage systems such as PASS [40,41] are interesting because they maintain both the data itself and the provenance trace for the generation of the data. One of the novel features of PASS is that it is able to generate executable scripts which describe the generation of a piece of data. These can then be applied to other data and the user can be sure that the same process has been followed. However they do not consider using the provenance and performance characteristics to optimise the cost of storing data.

7. Conclusions

An ideal cloud workflow environment would be able to accurately predict how long a given operation would take, the exact volumes of data generated, be able to recreate any piece of intermediate data on demand and do so at the lowest possible cost. Whilst there is no system available that can perform all of these tasks, techniques are available that can, to a degree, accomplish some of these feats.

This paper has demonstrated that if, in addition to the storage of basic provenance data, performance information such as execution times, data volumes and cloud operational costs are captured, it is feasible to approximate execution times and to develop long term data storage strategies that can yield significant cost savings.

By adopting a set of simple approximations and fall-back strategies, we have also demonstrated that even with very limited performance data it is still possible, to a degree, to obtain computation and storage cost estimates that can be used to inform storage strategies. As more data is collected, these approximations can be continuously improved and used to update execution models and refine storage strategies over time.

The field of medical data processing is an interesting one as it exercises the need for a comprehensive system for storing and using provenance. At a fundamental level, provenance is critical if audibility requirements are to be met whilst at a higher level, the potential cost savings associated with optimising the storage of intermediate results is significant.

Acknowledgements

The authors wish to thank Professor Mike Trennel and Dr. Vincent van Hees for their explanations of movement science and Dr. Paolo Missier for sharing his expertise on provenance modelling.

References

[1] I. Altintas, M.K. Anand, D. Crawl, A. Belloum, P. Missier, C. Goble, P. Sloot, Understanding collaborative studies through interoperable workflow provenance, in: Procs. IPAW 2010, Troy, NY, 2010.
[2] P. Missier, B. Ludascher, S. Bowers, M.K. Anand, I. Altintas, S. Dey, A. Sarkar, B. Shrestha, C. Goble, Linking multiple workflow provenance traces for interoperable collaborative science, in: Proc.s 5th Workshop on Workflows in Support of Large-Scale Science (WORKS), 2010.
[3] S. Woodman, H. Hiden, P. Watson, P. Missier, Achieving reproducibility by combining provenance with service and workflow versioning, in: Proceedings of the 6th Workshop on Workflows in Support of Large-Scale Science, WORKS'11, ACM, New York, NY, USA, 2011, pp. 127–136.
[4] J. Cala, E. Marei, Y. Xu, K. Takeda, P. Missier, Scalable and efficient whole-exome data processing using workflows on the cloud, Future Gener. Comput. Syst. (2016).
[5] S. Sabia, P. Cogranne, V.T. van Hees, J.A. Bell, A. Elbaz, M. Kivimaki, A. Singh-Manoux, Physical activity and adiposity markers at older ages: Accelerometer vs questionnaire data, J. Am. Med. Dir. Assoc. 16 (5) (2015) 438.e7–438.e13.
[6] S.E. Crouter, J.R. Churilla, D.R. Bassett, Estimating energy expenditure using accelerometers, Eur. J. Appl. Physiol. 98 (6) (2006) 601–612.
[7] S. Zhang, A. Rowlands, P. Murray, T. Hurst, Physical activity classification using the genea wrist-worn accelerometer, Med. Sci. Sports Exerc. 44 (4) (2012) 742–748.
[8] V.T. van Hees, L. Gorzelniak, E.C. Dean Leon, M. Eder, M. Pias, S. Taherian, U. Ekelund, F. Renstrom, P.W. Franks, A. Horsch, S. Brage, Separating movement and gravity components in an acceleration signal and implications for the assessment of human daily physical activity, PLoS One 8 (4) (2013) e61691.
[9] U. Dayal, M. Castellanos, A. Simitsis, K. Wilkinson, Data integration flows for business intelligence, in: Proceedings of the 12th International Conference on Extending Database Technology: Advances in Database Technology, EDBT'09, 2009, pp. 1–11.
[10] T. Heinis, G. Alonso, Efficient lineage tracking for scientific workflows, in: Proceedings of the 2008 ACM SIGMOD Conference, 2008, pp. 1007–1018.
[11] M.K. Anand, S. Bowers, T.M. McPhillips, B. Ludäscher, Exploring scientific workflow provenance using hybrid queries over nested data and lineage graphs, in: SSDBM, 2009, pp. 237–254.
[12] D.D. Roure, K. Belhajjame, P. Missier, J.M. Gomez-Perez, R. Palma, J.E. Ruiz, K. Hettne, M. Roos, G. Klyne, C. Goble, Towards the Preservation of Scientific Workflows, iPress, 2011.

Please cite this article in press as: S. Woodman, et al., Applications of provenance in performance prediction and data storage optimisation, Future Generation Computer Systems (2017), http://dx.doi.org/10.1016/j.future.2017.01.003

[13] H. Hiden, S. Woodman, P. Watson, J. Cala, Developing cloud applications using the e-science central platform, Phil. Trans. R. Soc. A 371 (1983).
[14] P. Missier, S. Woodman, H. Hiden, P. Watson, Provenance and data differencing for workflow reproducibility analysis, Concurr. Comput.: Pract. Exper. 28 (4) (2016) 995–1015.
[15] L. Moreau, et al., The open provenance model — core specification (v1.1), Future Gener. Comput. Syst. 7 (21) (2011) 743–756.
[16] D. Dominguez-Sal, P. Urbón-Bayes, A. Giménez-Vañó, S. Gómez-Villamor, N. Martínez-Bazán, J.L. Larriba-Pey, Survey of graph database performance on the HPC scalable graph analysis benchmark, in: Proceedings of the 2010 International Conference on Web-age Information Management, WAIM'10, Springer-Verlag, Berlin, Heidelberg, 2010, pp. 37–48.
[17] S. Woodman, H. Hiden, M. Turner, S. Dowsland, P. Watson, Monitoring of upper limb rehabilitation and recovery after stroke: An architecture for a cloud-based therapy platform, in: 2015 IEEE 11th International Conference on e-Science (e-Science), 2015, pp. 381–390.
[18] P. Watson, H.G. Hiden, S.J. Woodman, D.E. Leahy, J. Cala, P. Missier, The panel of experts cloud pattern, in: Proceedings of the third international workshop on Cloud data management, CloudDB'11, ACM, New York, NY, USA, 2011, pp. 23–24.
[19] H. Hiden, S. Woodman, P. Watson, A framework for dynamically generating predictive models of workflow execution, in: Proceedings of the 8th Workshop on Workflows in Support of Large-Scale Science, WORKS'13, ACM, New York, NY, USA, 2013, pp. 77–87.
[20] K. Wolstencroft, et al., The taverna workflow suite: designing and executing workflows of web services on the desktop, web or in the cloud, Nucleic Acids Res. 41 (W1) (2013) W557–W561.
[21] S. Woodman, H. Hiden, P. Watson, Workflow provenance: An analysis of long term storage costs, in: Proceedings of the 10th Workshop on Workflows in Support of Large-Scale Science, WORKS'15, ACM, New York, NY, USA, 2015, pp. 9:1–9:9.
[22] C. Walter, Kryder's law, Sci. Am. 293 (2005) 32–33.
[23] Y.L. Simmhan, B. Plale, D. Gannon, A survey of data provenance in e-science, SIGMOD Rec. 34 (3) (2005) 31–36.
[24] J. Freire, D. Koop, E. Santos, C. Silva, Provenance for computational tasks: A survey, Comput. Sci. Eng. 10 (3) (2008) 11–21.
[25] K. Belhajjame, et al., Why workflows break: Understanding and combating decay in taverna workflows, in: E-SCIENCE'12, IEEE Computer Society, Washington, DC, USA, 2012, pp. 1–9.
[26] D. de Oliveira, F. Costa, V. Silva, K. Ocaña, M. Mattoso, Debugging scientific workflows with provenance: Achievements and lessons learned, in: 29th SBBD–SBBD Proceedings, Curitiba, PR, Brazil.
[27] R. Jagadeesan, A. Jeffrey, C. Pitcher, J. Riely, Towards a Theory of Accountability and Audit, 2009, pp. 152–167.
[28] D. Hull, K. Wolstencroft, R. Stevens, et al., Taverna: a tool for building and running workflows of services, Nucleic Acids Res. 34 (suppl 2) (2006) W729–W732.
[29] B. Ludäscher, I. Altintas, C. Berkley, Scientific workflow management and the Kepler system, Concurr. Comput.: Pract. Exper. (2005) 1039–1065.
[30] P. Missier, S. Sahoo, J. Zhao, C. Goble, A. Sheth, Janus: From workflows to semantic provenance and linked open data, in: Provenance and Annotation of Data and Processes, in: Lecture Notes in Computer Science, vol. 6378, Springer, Berlin, Heidelberg, 2010, pp. 129–141.
[31] P. Missier, N. Paton, K. Belhajjame, Fine-grained and efficient lineage querying of collection-based workflow provenance, in: Procs. EDBT, Lausanne, Switzerland, 2010.
[32] H. Wendel, M. Kunde, A. Schreiber, Provenance of software development processes, in: Provenance and Annotation of Data and Processes, in: Lecture Notes in Computer Science, vol. 6378, Springer, Berlin / Heidelberg, 2010, pp. 59–63.
[33] G. Tylissanakis, Y. Cotronis, Data provenance and reproducibility in grid based scientific workflows, Grid and Pervasive Computing Conference, Workshops at the 0, 2009, pp. 42–49.
[34] R. Duan, F. Nadeem, J. Wang, Y. Zhang, R. Prodan, T. Fahringer, A hybrid intelligent method for performance modeling and prediction of workflow activities in grids, in: 9th IEEE/ACM International Symposium on Cluster Computing and the Grid, 2009. CCGRID'09. 2009, pp. 339–347.
[35] R. Cushing, S. Koulouzis, A.S.Z. Belloum, M. Bubak, Prediction-based auto-scaling of scientific workflows, in: Proceedings of the 9th International Workshop on Middleware for Grids, Clouds and e-Science, MGC'11, ACM, New York, NY, USA, 2011, pp. 1:1–1:6.
[36] N. Roy, A. Dubey, A. Gokhale, Efficient autoscaling in the cloud using predictive models for workload forecasting, in: 2011 IEEE International Conference on Cloud Computing (CLOUD), 2011, pp. 500–507.
[37] V. Taylor, X. Wu, J. Geisler, R. Stevens, Using kernel couplings to predict parallel application performance, in: 11th IEEE International Symposium on High Performance Distributed Computing, 2002. HPDC-11 2002. Proceedings., 2002, pp. 125–134.
[38] S. Sadjadi, S. Shimizu, J. Figueroa, R. Rangaswami, J. Delgado, H. Duran, X. Collazo-Mojica, A modeling approach for estimating execution time of long-running scientific applications, in: IEEE International Symposium on Parallel and Distributed Processing, 2008. IPDPS 2008., 2008, pp. 1–8.
[39] V. Cuevas-Vicenttín, P. Kianmajd, B. Ludäscher, P. Missier, F. Chirigati, Y. Wei, D. Koop, S. Dey, The PBase scientific workflow provenance repository, in: The International Journal of Digital Curation, Vol. 9, San Francisco, CA, USA, 2014, pp. 28–38.
[40] K.-K. Muniswamy-Reddy, D.A. Holland, U. Braun, M. Seltzer, Provenance-aware storage systems, in: USENIX'06, Berkeley, CA, USA, 2006, pp. 43–56.
[41] K.-K. Muniswamy-Reddy, et al. Layering in provenance systems, in: USENIX'09, Berkeley, CA, USA, 2009, pp. 10–10.

Simon Woodman is a Senior Research Associate in the Digital Institute and one of the architects of e-Science Central, a cloud-based system to support data storage and analytics within the scientific domain. He has worked extensively on both the core of the system and the provenance capture component. e-Science Central is used extensively in both academic and commercial settings. As the technical lead of the MOVEeCloud project, Simon oversees the use of e-Science Central for storing and analysing Physical Activity data. Currently 3 ongoing physical activity studies are using e-Science Central to store their research data and analyse and share the results. Another project Simon is heavily involved in, Limbs Alive, is using e-Science Central to support the rehabilitation of stroke victims using computer games.

As a member of the Digital Institute, Simon is used to conducting interdisciplinary research and helping a diverse set of researchers improve the quality of their work through the application of technology. In the past, Simon has worked on projects including Building Information Modelling, Spectral Analysis, Flood Modelling, Light Rail Infrastructures and Social Exclusion. He regularly speaks on Cloud Computing, Big Data and Data Analytics.

As part of Project Junior, Simon helped researchers from the Northern Institute of Cancer Research to build 1M models of chemical structure–activity data. This work was continued in the EU FP7 project, VENUS-C, where the functionality was migrated to Windows Azure.

Simon was awarded his Ph.D. in 2008 for his thesis on "A Programming System for Process Coordination in Virtual Organisations", supervised by Prof. Santosh Shrivastava.

Simon has been involved with many projects in the areas of distributed systems, workflow, service description, e-Science and cloud computing. He was responsible for the service orchestration aspects of the ADAPT project, developing a decentralised workflow engine for Web Services. With colleagues he was involved with the creation of the SOAP Services Description Language, SSDL, which offered a contractual description of Web Services. The SSDL-Sequencing Constraints framework offered a formal model based on π-calculus of the interaction between web services.

Simon has experience in usability having worked in the User Centered Design group at IBM Hursley during an internship as part of his undergraduate course. He was one of the developers of the original CICS Information Centre, presenting the CICS documentation in novel ways.

Hugo Hiden is Senior Research Associate in the School of Computing Science at Newcastle University. His current role is as the software development lead on the SiDE project, which is a £12M five year project aiming to tackle social exclusion by making it easier for people to access the life-changing benefits offered by digital technologies. He is also the principal developer of the e-Science Central platform, a cloud based data processing system which has been used in a number of academic and commercial contexts. Prior to his University role, he spent 6 years in industry developing advanced data integration, analysis and modelling tools at Avantium Technologies and GSE Systems Inc. Hugo holds a Ph.D. in the application of Genetic Programming to chemical process data analysis and has published numerous papers on the subjects of cloud computing, process monitoring, computer security and collaborative R&D.

Paul Watson is Professor of Computer Science and Director of the Digital Institute. He is PI of the EPSRC Centre for Doctoral Training in Cloud Computing for Big Data and also directed the £12M RCUK-funded Digital Economy Hub on Social Inclusion through the Digital Economy. He graduated in 1983 with a B.Sc. in Computer Engineering from Manchester University, followed by a Ph.D. on parallel graph reduction in 1986. In the 80s, as a Lecturer at Manchester University, he was a designer of the Alvey Flagship and Esprit EDS systems. From 1990 to 5 he worked for ICL as a system designer of the Goldrush MegaServer parallel database server, which was released as a product in 1994.

In August 1995 he moved to Newcastle University, where he has been an investigator on research projects worth over £40M. His research interest is in scalable information management with a current focus on Cloud Computing. He sits on the board of Dynamo North East, an industry-led organisation created to grow the IT economy of the region. Professor Watson is a Chartered Engineer, a Fellow of the British Computer Society, and a member of the UK Computing Research Committee. He received the 2014 Jim Gray eScience Award.

Please cite this article in press as: S. Woodman, et al., Applications of provenance in performance prediction and data storage optimisation, Future Generation Computer Systems (2017), http://dx.doi.org/10.1016/j.future.2017.01.003